Consensus Across Continents

Benjamin Bengfort, Rebecca Bilbro, Pete Keleher
Department of Computer Science
University of Maryland, College Park, MD, USA
{bengfort,rbilbro,keleher}@cs.umd.edu

Abstract—**Distributing data storage systems across wide geographic areas provides resilience to catestrophic failure and improves performance by localizing user access. However, as network distance increases, the impact of failure modes such as partitions and communication variability pose significant challenges to coordination that impair strong consistency, particularly when systems scale beyond a handful of replicas. In order to balance consistency and performance in a multi-region context, geo-distributed consensus must be flexible, adapting to changing network conditions and user behavior. In this paper, we introduce Alia, a hierarchical consensus protocol that implements and extends Vertical Paxos, designed to implement large, strongly-consistent and adaptable geo-replicated consensus groups. Alia splits coordination responsibility across two tiers: a root quorum responsible for safely moving the system through reconfigurations, and subquorums that manage accesses. Subquorums intersect with the root quorum using a novel method, delegated voting, which ensures that all replicas participate in both consensus tiers and provide transparent, linearizable guarantees across the entire system. This design ensures Alia can optimize throughput and availability by flexibly changing its configuration in real-time to meet demand without sacrificing consistency.**

Index Terms—**hierarchical consensus, geographic replication, delegated voting, strong consistency**

I. Introduction

The recent availability of cloud service providers with data centers that span the globe has made it easier than ever before to deploy geographically distributed data systems that span continents and oceans. These types of systems increase local performance by minimizing network distance between users and replicas and provide the opportunity for data recovery in the face of catastrophes such as floods or earthquakes. Moreover, the success of specialized, high-availability data systems [2], [3], [5], [9] in maximizing throughput across the wide area has led to increased interest in geo-replicated systems, particularly as more applications are being developed with international audiences in mind. However, in order to generalize these systems for modern application development, strong consistency semantics is required; therefore managed replicated data services [1], [7], [19] have risen to prominence, achieving these semantics by hiding both the replication and infrastructure complexity from developers.

At the same time, traditional monolithc applications are being replaced by microservice architectures and cloud-native service meshes [10] that make infrastructure directly visible to applications. As applications scale, service meshes make it easier to maintain and optimize service-specific communication to minimize downtime and to improve system flexibility. Additionally, due to increasing privacy regulation, application developers require more control over data placement rather than less [17]. The engineering-based solutions of managed geo-distributed data services are designed to coordinate hundreds of replicas that have access to expensive data-center hardware and involves multiple, independent processes and quorums to synchronize time, allocate locks, manage transations, and recover from failure. Although these systems provide strong consistency, they do so in an rigid, opaque manner that is not flexible enough for developers who require strong consitency at a higher level of the application stack.

We propose a simpler approach to building large, geographically replicated systems. Rather than relying on a fleet of loosely-coupled, independent small quorums whose interactions are difficult to reason about, we propose a single, system-wide consensus protocol that coordinates both replica placement and data accesses. By ensuring that all coordination occurs through a single consensus activity, it is easier to reason about the consistency of the system even in a network environment prone to correlated failures, partitions, and variable latency. Additionally, a single source of coordination gives the system the freedom to adapt to changes in access patterns, configure to maximize throughput, specify data placement rules, and ensure straightforward system maintenance.

In order to achieve this, a new consensus protocol that can scale beyond a handful of replicas is required. Distributed consensus, canonically represented by Paxos [14] and its performance optimizing variants [4], [6], [12], [13], primarily consider safety in the case of one or two fail-stop node failures. Although some recent research has explored the problem of geo-distributed consensus [15], [16], it primarily considers the problem of high-latency links but geo-replication implies scale. Services running around the globe recquire dozens if not hundreds of replicas and introduce new failure modes such as network partitions, where sections of the system operate independently without fail-stop failure, and highly variable latency that inhibit quorum progress. In order to scale systems beyond a handful of replicas, current systems [7], [8], [11], [18] use Paxos as a component, instantiated across multiple transactions, shards, or tablets to manage small subsystems independently, leading to increased complexity and reduced transparency.

We introduce a novel approach to scale consensus beyond a handful of nodes: *hierarchical consensus*. Our approach is to similarly decompose the consensus problem into units that can be handled by provenly safe algorithms, but organizes all managed processes into an intersecting hierarchy of quorums that ensure that all system-wide consensus decisions are totally ordered. The challenge is in building a multi-group coordination protocol that configures and mediates subquorums through a root quorum. The root quorum gaurantees correctness by pivoting the system through reconfigurations that place replicas into subquorums and maps them to partitions of the object namespace to handle direct data accesses.

BJB: ended here

The root quorum is composed of all replicas in the system, although reconfigurations are rare with respect to data accesses, we introduce *delegated voting* to optimize quorum decisions at the root.

Much of the systems complexity comes from handshaking between the root quorum and subquorums during reconfiguration. These handshakes are made easier and far more efficient by using *fuzzy transitions*, which allow individual subquorums to move through reconfiguration at their own pace without

We validate our approach by implementing hierarchical consensus in Alia, a linearizable object store explicitly intended to run with many replicas, geo-replicated across heterogenous networks and devices. The resulting system is local, in that replicas serving clients can be located near them. The system is fast because individual operations are served by a small group of replicas regardless of the size of the total system. The system is nimble in that it it can dynamically reconfigure the number, membership, and responsibilities of the subquorums in response to failures, phase changes in the driving applications or policy requirements for data placement and durability. Finally, the system is consistent, supporting the strongest form of per-object consistency without relying on special-purpose hardware. We demonstrate its advantages through an implementation scaling to hundreds of replicas across more than a dozen availability zones around the world using Amazon EC2.

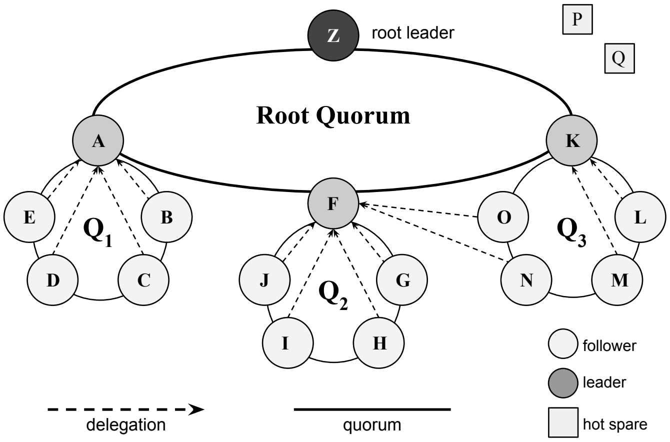

Fig. 1. Replicas participate in intersecting tiers of consensus.

II. Conclusion

Future work: investigate more tiers

References

[1] CockroachDB Geo-Partitioning.

[2] Muthukaruppan Annamalai, Kaushik Ravichandran, Harish Srinivas, Igor Zinkovsky, Luning Pan, Tony Savor, David Nagle, and Michael Stumm. Sharding the shards: Managing datastore locality at scale with Akkio. In *13th ${$USENIX$}$ Symposium on Operating Systems Design and Implementation (${$OSDI$}$ 18)*, pages 445–460.

[3] Jason Baker, Chris Bond, James C. Corbett, J. J. Furman, Andrey Khorlin, James Larson, Jean-Michel Leon, Yawei Li, Alexander Lloyd, and Vadim Yushprakh. Megastore: Providing Scalable, Highly Available Storage for Interactive Services. In *CIDR*, volume 11, pages 223–234.

[4] Martin Biely, Zoran Milosevic, Nuno Santos, and Andre Schiper. S-paxos: Offloading the leader for high throughput state machine replication. In *Reliable Distributed Systems (SRDS), 2012 IEEE 31st Symposium On*, pages 111–120. IEEE.

[5] Nathan Bronson, Zach Amsden, George Cabrera, Prasad Chakka, Peter Dimov, Hui Ding, Jack Ferris, Anthony Giardullo, Sachin Kulkarni, and Harry Li. Tao: Facebook's Distributed Data Store for the Social Graph. In *Presented*

as Part of the 2013 USENIX Annual Technical Conference (USENIX ATC 13), pages 49–60.

[6] Lásaro Jonas Camargos, Rodrigo Malta Schmidt, and Fernando Pedone. Multicoordinated paxos. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Principles of Distributed Computing*, pages 316–317. ACM.

[7] James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, J. J. Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, et al. Spanner: Google's globally distributed database. 31(3):8.

[8] Lisa Glendenning, Ivan Beschastnikh, Arvind Krishnamurthy, and Thomas Anderson. Scalable consistency in Scatter. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, pages 15–28. ACM.

[9] Sudarshan Kadambi, Jianjun Chen, Brian F. Cooper, David Lomax, Raghu Ramakrishnan, Adam Silberstein, Erwin Tam, and Hector Garcia-Molina. Where in the world is my data. In *Proceedings International Conference on Very Large Data Bases (VLDB)*.

[10] Matt Klein. Lyft's Envoy: Experiences Operating a Large Service Mesh.

[11] Tim Kraska, Gene Pang, Michael J. Franklin, Samuel Madden, and Alan Fekete. MDCC: Multi-Data Center Consistency. In *Proceedings of the 8th ACM European Conference on Computer Systems*, pages 113–126. ACM.

[12] Leslie Lamport. Fast paxos. 19(2):79–103.

[13] Leslie Lamport. Generalized consensus and Paxos.

[14] Leslie Lamport. Paxos made simple. 32(4):18–25.

[15] Yanhua Mao, Flavio Paiva Junqueira, and Keith Marzullo. Mencius: Building efficient replicated state machines for WANs. In *OSDI*, volume 8, pages 369–384.

[16] Iulian Moraru, David G. Andersen, and Michael Kaminsky. There is more consensus in egalitarian parliaments. In *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, pages 358–372. ACM.

[17] Aashaka Shah, Vinay Banakar, Supreeth Shastri, Melissa Wasserman, and Vijay Chidambaram. Analyzing the Impact of GDPR on Storage Systems. In *11th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage 19)*.

[18] Alexander Thomson and Daniel J. Abadi. CalvinFS: Consistent wan replication and scalable metadata management for distributed file systems. In *13th USENIX Conference on File and Storage Technologies (FAST 15)*, pages 1–14.

[19] Alexandre Verbitski, Anurag Gupta, Debanjan Saha, Murali Brahmadesam, Kamal Gupta, Raman Mittal, Sailesh Krishnamurthy, Sandor Maurice, Tengiz Kharatishvili, and Xiaofeng Bao. Amazon aurora: Design considerations for high throughput cloud-native relational databases. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1041–1052. ACM.